440-TP-005-001

Physical Access & Media Management for the ECS Project

Technical Paper

Technical Paper—Not intended for formal review or government approval.

February 1995

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

Tom Smith /s/	2/28/95
Tom Smith, Senior Systems Engineer	Date
EOSDIS Core System Project	

SUBMITTED BY

Stephen Fox /s/	3/18/95
Steve Fox, SDPS Segment Manager	Date
EOSDIS Core System Project	

Hughes Applied Information Systems
Landover, Maryland

This page intentionally left blank.

Contents

Appendix B

Appendix C

Abbreviations and Acronyms

1. Introduction

1.1 Purpose

The concepts of Physical Access and Media Management (PAMM) for the vast amount of data that will reside in ECS is a matter concern to NASA and the Earth Science Community. The current generation of File Storage Management System (FSMS) software products, provide a majority of the functionality required by ECS. Unfortunately, there is no single product that will meet all requirements for the Data Server Architecture. In addition, a single system will not be able to provide sufficient bandwidth to manage the entire storage architecture, and service all requests in a timely manner. This results in the need for some form of distributed storage system that can address the scalability, evolvability, cost, and performance requirements of ECS.

1.2 Scope

This working paper focuses on the architectural issues of physical access to data and management of the associated media with regard to scalability, evolvability, cost, and performance. This paper will identify a basic set of requirements and enumerate several candidate implementations in general terms. Further analysis and architectural work is required to validate the more viable implementation options.

1.3 Organization

This paper will begin with a general discussion of concepts surrounding data and media management and various methods by which scalability can be achieved today. A brief discussion of selected governmental, educational, and industrial sites that have similar problems and their present solutions will follow. Finally, potential configurations for ECS solutions will be listed along with general risk assessments for each approach. Specific requirements for PAMM components will be provided in a series of appendices.

This page intentionally left blank.

2. Data & Media Management

2.1 Concepts

The majority of the storage systems in production today can be logically divided into two general parts. (1) File Management - which entails tracking a file throughout its lifetime within a system and providing the file to authorized users upon demand. The file manager stores the attributes of each data file and knows which medium or media each file is resident on. (2) Media Management - which consists of managing the physical media volumes in the associated manual and/or automated storage libraries. These libraries may consist of multiple form factors of media, S-VHS, 3480, 5.25 inch Magneto-Optical disks, etc. The media manager knows the physical location and form factor of each volume in the library.

2.2 Data Management

Many production systems in use today utilize a single host approach that combines the capabilities of Data and Media Management. Files are managed on the storage available to the host and media is managed via the data interfaces available to the host. The majority of these systems utilize some form of Hierarchical Storage Management (HSM) technology. Frequently accessed, high interest data, is keep on-line on fast media (solid state disk, RAID, magnetic disk, etc.) As the data becomes less interesting, it is moved to slower media types (optical disks, magnetic tapes, etc.) Eventually, this data may be placed off-line in a manual library.

The single host approach (See Figure 2-1) is sufficient for systems if the following conditions can be met. (1) The system either has a predictable or slow growth pattern (e.g. a file backup system). (2) The amount of data in the active system[1] is or can be limited. (3) The number of users and/or queries against data in the system is limited. A system that cannot meet these conditions will eventually suffer performance problems in a single host environment. Normally, the system administrator and/or the data center must artificially (externally) control the storage system to minimize performance problems (e.g. Batch mode retrievals from a backup system). The single host approach limits both scalability and evolvability.

Some storage systems such as the NASA Center for Computational Science (NCCS) are experiencing a tremendous growth in data storage requirements while also experiencing growing user demand. This has resulted in a large super computer (Convex 38xx) being tasked to manage the data, reduced system response time, and the need for off-line media storage. Many industrial concerns such as Mobil Oil require vast amounts of data storage to provide seismic data to the complex models used to determine optimal locations for exploration platforms. Much of this data must be held off-line until needed. Since an single exploratory oil well may cost upwards of sixty million dollars, accurate processing and storage of data is a major concern.

[1] In this case the term active system is synonymous with on-line system. Data that is stored in a manual archive is considered off-line. An example of this is a payroll system. The current year's transactions are on-line with pervious years transactions off-line.

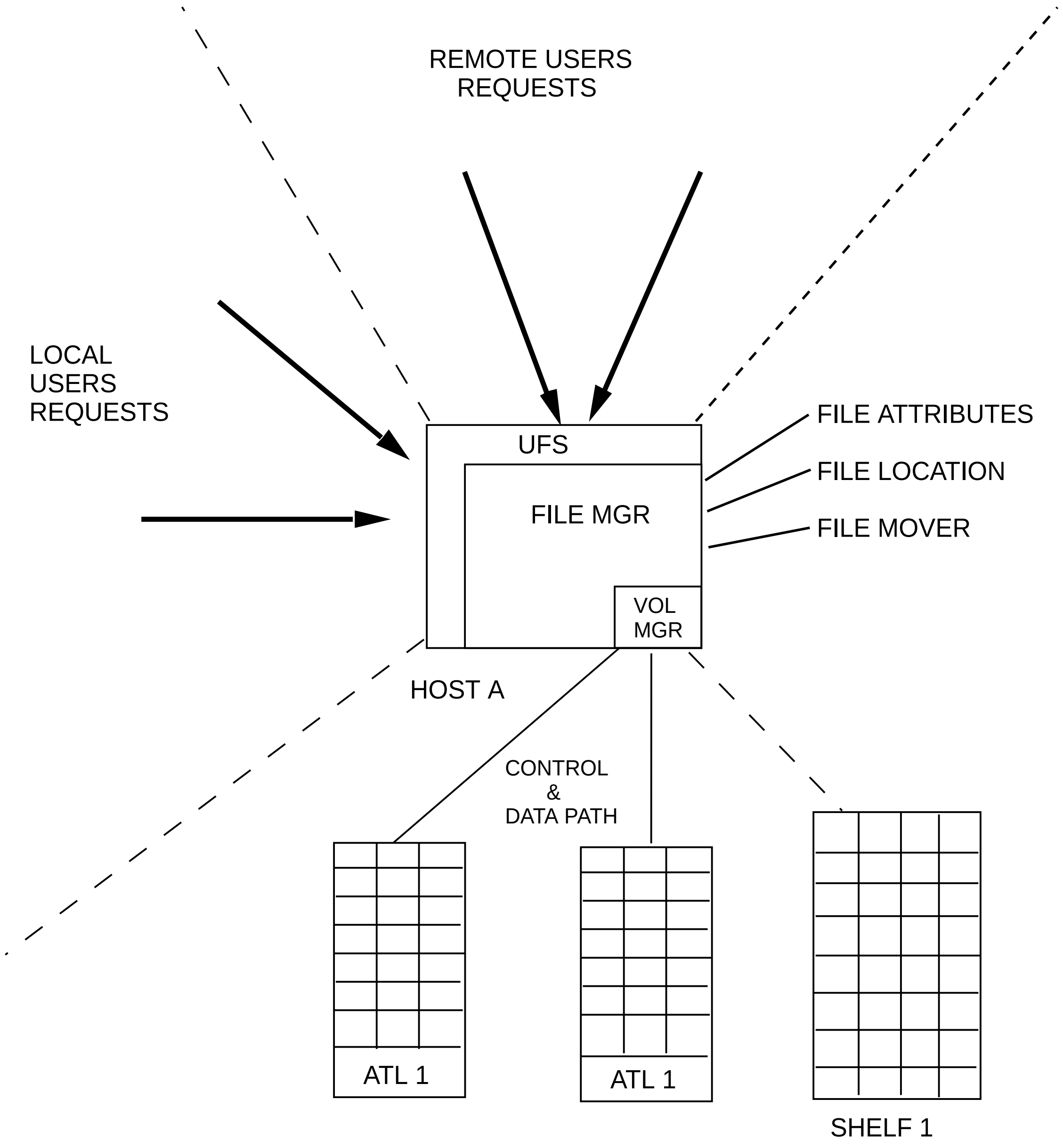

Figure 2-1. Today's General FSMS Configuration

2.3 Physical Access To Data

The primary purpose of a data storage system is to provide authorized users with accurate data in a timely manner. There are two major ways that data may be accessed by users. (1) Local access is a term that denotes the ability of the user to login to the storage management host. This is either with a direct connect terminal or via telnet. Once on the host the user may access files explicitly if they reside in memory or on disk. If the files have migrated from disk (as in an HSM system) the user may still access these files explicitly via the query scheme provided by the storage product. (e.g. SQL or product supplied retrieval commands.) An explicit command causes data to be moved from tertiary storage back to secondary storage. Data may be accessed implicitly (again, in an HSM system) by performing a Unix operation on the associated file. (e.g. a move or edit operation will cause the requisite file to be copied from tertiary to secondary storage.) (2) Remote access to data means the user does not need to login to the storage host to access the stored data. Remote access is normally associated with access via a Local Area or Wide Area Network (LAN/WAN). There are a number of methods by which this is achieved. These are:

afs	Andrew File System
dfs	Distributed File System
nfs	Network File System
ftp	File Transfer Protocol
rcp	Remote Copy
rpc	Remote Procedure Calls
uucp	Unix to Unix Copy

The first three methods, *afs, dfs, nfs* represent remote file system environments. These environments give a user a better view into the contents of the storage system. Users may browse directories of files (assuming they have the appropriate file privileges) and access files they find interesting. Each of these methods has certain costs (performance, manageability, synchronization, etc.), benefits (near transparent remote access, simulated local file access, etc.), and maturity levels (availability of the product on specific hosts, size of the user community, etc.) associated with it. The last three methods *ftp, rcp, rpc* are primarily designed to retrieve specific files.

A remote file system environment is a key component of the ECS. The selection criteria for this component includes, scalability, performance, manageability, evolvability and cost. General requirements for a remote file system are attached in Appendix A.

This page intentionally left blank.

3. Alternate Architectures

3.1 Separating Data & Media Management

One method of increasing performance and scalability is to separate the general functions of file and media management (General Requirements in Appendix B and Appendix C). This allows one host to manage user requests and the file data. Another host manages the location of the media, status of the available drives, and mounting of media on demand. This reduces the overall load on a single CPU and provides for some improvement in data transfer rates since the control and data paths can be separated (See Figure 3-1). The following Volume Management products are known to exist now.

Volume Manager	EMC - Epoch Inc.	Beta Test - Component of a larger package - Company is undecided if it will be marketed separately. If so, it will be an OEM Product only
TYEE	Digital Equipment Corp.	Beta Test - OEM Product Initially
REELlibrarian	CRAY Research	Available Today - Cray Systems Only
VolServ	EMASS Inc.	Available Today - Full Function

Two of these products are immature and do not, as yet, have a user base that can be consulted about operational stability and problems. REELlibrarian is available today but it lacks APIs and RPC libraries to allow rapid integration of new technologies. In addition, this product only runs on Cray systems. Only the VolServ product from EMASS Inc., (used by Exxon Corp. in Dallas and U.S. Army Corps of Engineers in Vicksburg, among other sites) appears to have the necessary maturity, product host diversity, evolvability, and APIs & RPC libraries to support the ECS architecture at this time.

A separated file and volume serving configuration is evolvable in that the file management component can be changed without affecting the stored volumes (data formats aside). A file meta-data transfer or translation must occur between the old and new file manager to insure the name of each file is associated with a named media volume. The volume serving component will retrieve all volumes based on each medium's unique name. This configuration is also scaleable since a single volume server can manage multiple storage devices for multiple file managers. Cost can be reduced by purchasing less powerful (and expensive) host platforms since software complexity (and thus, CPU utilization) has also been reduced.

This configuration could be further exploited at the local site with the introduction of network addressable storage devices and direct-to-to client distribution via a high speed data network (See Figure 3-2). This configuration adds additional fault tolerance and maximizes throughput for archive to local user transfers.

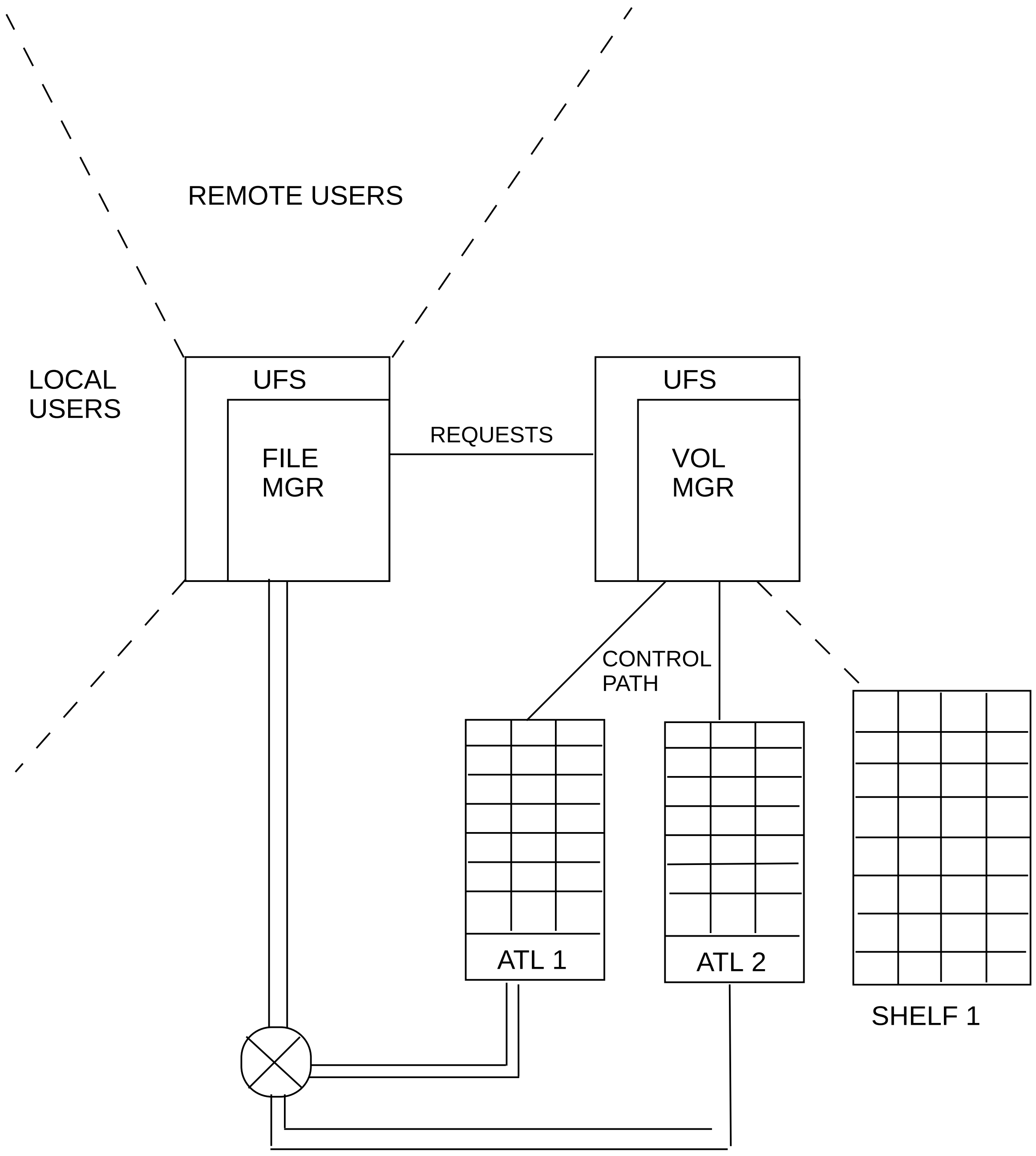

Figure 3-1. Separation of File & Media Management

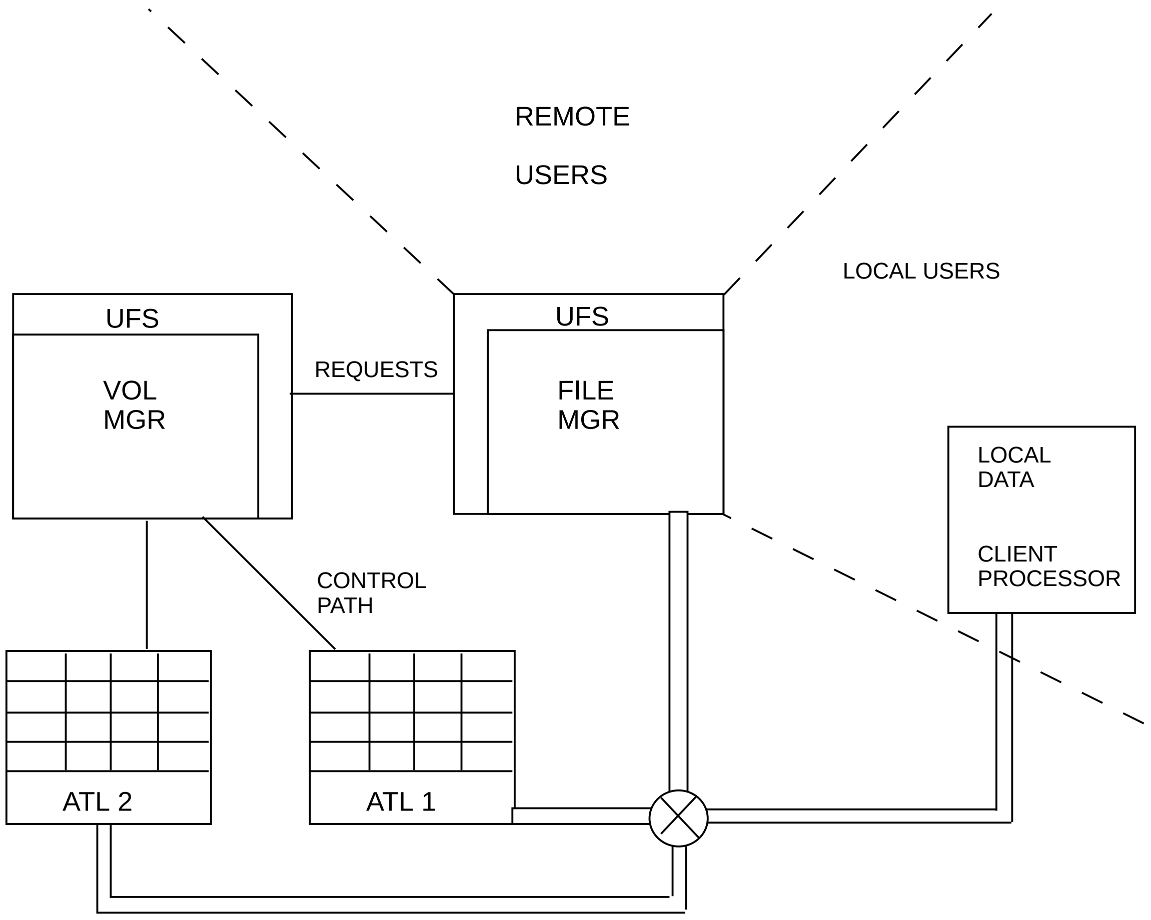

Figure 3-2. Separation of Control and Data Streams

3.2 Exporting the Namespace

A large percentage of the requests entering ECS will be generated from external entities. Requests from other DAAC sites, SCFs, researchers, and World Wide Web users will form the bulk of the ECS data requirements. The archive must be able to meet these data requests in a timely manner. In addition, these requests will be coming from a variety of diverse host computers. A key step in requesting ECS data is identifying the file name that contains the requisite data. Searching the file name space will require some form of global namespace that can be accessed via a heterogeneous network.

Three methods of achieving a global namespace are currently possible. These are via *nfs, afs,* or *dfs*. Open Software Foundation's *Distributed File System* is still in its infancy. This product is closely integrated with the Distributed Computing Environment (DCE) and is slowly evolving to meet specific user needs and requirements. This maturing process will require several years, therefore *dfs* is not a near-term solution.

This leaves the Network File System (*nfs*), perhaps the most widely used method in the world, and the Andrew File System *(afs)*. Ongoing prototypes and experience with both *nfs* & *afs* as well as experiences by other sites such as the National Center for Supercomputing Applications (NCSA) [Yeager, et al.] indicate that administrative simplicity, scalability, and performance are all enhanced by *afs*. Translation tools are offered for *nfs* to *afs* queries.

The selection of the appropriate network file system is out of scope for this document, however, the incorporation of a technique for managing the global namespace will impact file & media management as well as data access. Utilizing a global namespace approach in conjunction with the separation of file and volume management offers several advantages. The complexity of the file management component of the system may be significantly reduced if file name and namespace management is layered into the network communications environment. Separate host servers would manage the name space. Loss of a name space server does not form a single point of failure. In addition, file server complexity could be significantly reduced. This would allow file server replication and thus eliminate another single point of failure.

3.3 Distributed Network Components

The optimal configuration for high availability, scalability, and performance is to distribute the component parts of a file manager and a volume manager to separate hosts. As additional components are needed, based on user load or some other criteria, additional Unix processes on additional hosts can be started to meet the demand. A custom implementation of this approach is the High Performance Storage System (HPSS) being developed at Lawrence Livermore Labs for the Department of Energy. The ECS architecture also follows this paradigm except COTS components are substituted for HPSS components. (See Figure 3-3)

This approach is highly flexible and scaleable. The distributed nature of the architecture will allow a larger number of inexpensive workstations/servers to be substituted for larger host computers.

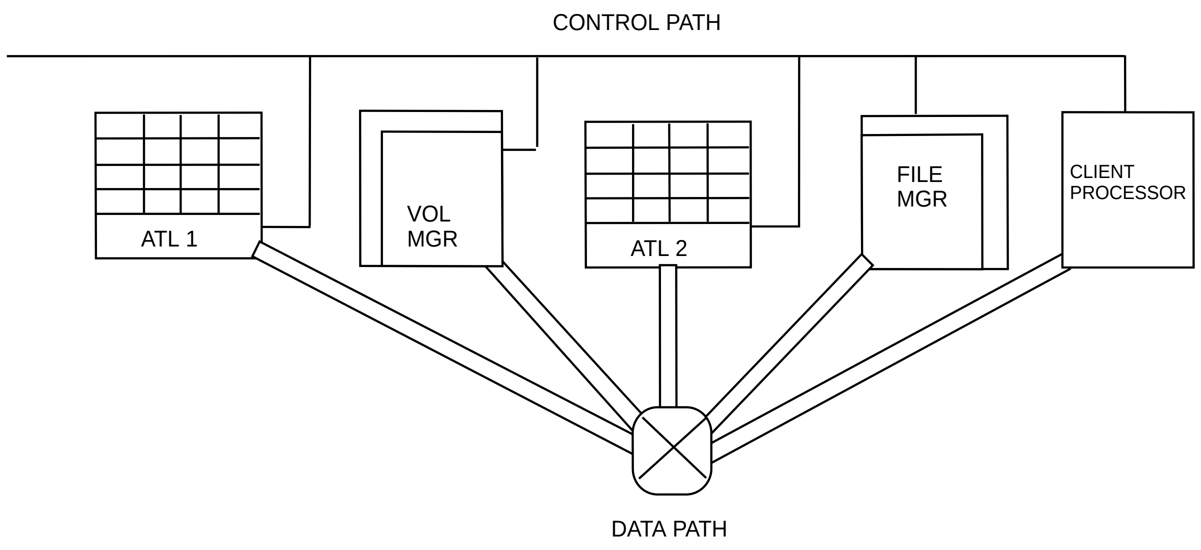

Figure 3-3. Networked Archive Components

This page intentionally left blank.

4. References

[Bach 86] Bach, Maurice J., The Design of the UNIX Operating System, Prentice Hall, Engelwood Cliffs, NJ, 1986.

[Bloomer 92] Bloomer, John, Power Programming with RPC, O'Reilly & Associates, Inc., Sebastopol, CA, 1992.

[Cray 92] Cray Research Inc., Cray/REELlibrarian (CRL) 1.0 Release Overview, RO-5205 1.0, 1992.

[Coyne 93] Coyne, R.A., H. Hulen and R.W. Watson, "The High Performance Storage System," Proceedings of Supercomputing 93, Portland, IEEE Computer Society Press, Nov. 1993.

[Digital 10/20/94] Digital Equipment Corp., "TYEE Product Brief", Non-Disclosure Meeting,

October 20, 1994.

[EMASS 93] EMASS Inc., "Volume Server Technical Summary" , E-Systems Technical Paper, 1993.

[EMASS 11/1/94] EMASS Inc., "VolServ: Technical Brief", Status Document, November 1, 1994.

[Epoch 10/12/94] EMC/Epoch Inc., "Windsor Architecture Briefing", Non-Disclosure Meeting, October 12, 1994.

[Epoch 10/27/94] EMC/EPOCH Inc., “Volume Manager: Technical Brief", Non-Disclosure Document, October 27, 1994.

[Goldlick 93] Goldlick, Jonathan S., K. Benninger, W. Brown, C. Kirby, C. Maher, D. Nydick, & B. Zumach, "An AFS-Based Supercomputing Environment," Proceedings of the Twelfth IEEE Symposium on Mass Storage Systems, Monterey, IEEE Computer Society Press, Apr. 1993. Associates, Inc., Sebastopol, CA, 1990.

[Smith 93] Smith, Thomas, "File Storage Management Systems Functionality And Performance White Paper For ECS," Draft 1.0, August 31, 1993.

[Stern 91] Stern, Hal, Managing NFS and NIS, O'Reilly & Associates, Inc., Sebastopol, CA, 1991.

[Stevens 90] Stevens, W. Richard, UNIX Network Programming, PTR Prentice Hall, Engelwood Cliffs, NJ, 1990.

[Transarc-SAG 92] Transarc Corporation, AFS System Administrator's Guide, FS-0200-00.10.3, July 1992.

[Yeager et al.]	Yeager, Nancy with Michelle Butler and Gerald Quinn, "Migration to the Andrew FileSystem [sic] at NCSA" Technical Paper, National Center for Supercomputing Applications, Jan. 1994.

Appendix A

Anticipated File System Requirements

1. **Support Shared Storage Devices.** - *A keystone of the SDPS Architecture presented at SDR is the idea that large, high speed, storage devices need not be replicated throughout the system. Instead these devices could be shared between hosts and sub-segments. The focus here was primarily sharing between DADS and PGS to avoid moving large files around the system.*

   - ***Shared Device Failover***. - There are several ways shared devices could be implemented. Any implementation must have a built in method to insure transfers to or from a device do not fail due to lack of space. e.g. Write retries to another shared device, pre-allocation of storage space, etc.

2. **Support Distributed Systems.**- *The ECS Architecture presented at SDR emphasizes the concept of close interactions between distributed sub-systems in a LAN environment. A network file system must be able to support several key concepts.*

   - ***File Security*** - *T*his consist of three tiers. 1) Transaction security - only authorized clients and servers can request ECS data. 2) Version security - unauthorized users must not be able to store new files or new versions of files on the ECS System. 3) File Security - unauthorized users must not be able to move, copy, modify or delete ECS data files.

   - ***Name Space Management*** - The distributed nature of the file system means that it must, to some extent, manage the name space of files under its control. Authorized users and processes must have some capability of finding physical locations of files within the file system. This may involve an additional external entity (i.e. RDBMS, flat file, etc.)

   - ***File Locking*** - The distributed file system must know when a file is open for modification and either provide another copy (read-only request) or notify the requester that the file is not available.

   - ***File Caching*** - Access to files should use the most efficient caching and access system possible. Benchmarks have shown that state-based communications approaches with file caching for read and write on the client are superior to stateless server cache based systems. Regardless of the method selected, efficiency and minimizing server involvement during client access is a prime goal here.

   - ***Volumes can exceed maximum partition size (2 Gigabytes).*** - A logical grouping mechanism to manage files and allow for file replication for load balancing must be provided. These volumes and individual files should be able to exceed 2 GB.

- ***SDPS File Object Size*** - At present, it is unclear how large a file object will be within SDPS and DADS. From a network perspective fewer, larger objects are more desirable than many smaller objects.

3. **Multiple File Residencies.** - *Multiple file residency addresses the DADS issue of storage tiers. There will likely be a pyramid of storage devices similar to the data pyramid. The top level is host memory or buffer space, the bottom will be some sequential access device.*

   - *File Residencies - The File System should allow for separate, equally accessible copies of the same file to reside at different levels of the storage pyramid. (e.g. a file is needed for hard media distribution to a scientist and for processing by PGS. The RAID copy of the file might be sent to PGS while the tape copy is used to create a hard media distribution copy.) Another example, is standing orders. A permanent copy of the data will be sent to tape, a disk copy is kept to process an imminent standing order.*

4. **Support for third party transfers.** - *A third party transfer can be interpreted in two ways. 1) Device to Device, bypassing host systems entirely. 2) Device to Application, bypassing system memory buffers and transferring data directly to the requesting applications in its memory space. SDPS & DADS are focusing on Method 1.*

   - ***Device to Device between Secondary Storage (Disk to Disk)***
   - ***Device to Device between Tertiary and Secondary Storage (Disk to Tape)***
   - ***Device to Device between Secondary and Tertiary Storage (Tape to Disk)***

5. **Provides an RPC-based mechanism for executing third party transfers.** - *Third party transfers can be initiated at three levels within a File System. The least invasive method is User Space - FTP or NFS. The next is an RPC based library of transfer commands and the last is with specific OS kernel modifications. Flexibility and portability issues coupled with functionality, control, and security considerations point to an RPC based implementation.*

   - ***Implementation focus on third party copy (device to device)***

6. **Supports a separation of Control & Data Transfer paths.** - *To assist in network balancing and to insure efficient utilization of resources as well as increase performance, the files system should support separate control (FIDDI, Ethernet, etc.) and high speed data transactions. (IPI-3, Fibre Channel, ATM, etc.)*

7. **Supports non-random access storage systems.** - *Any file system solution involving the data sever will require management or at least interaction with sequential access devices, i.e. tapes and their associated robotics. This is particularly true of third party transfers.*

8. **All allocation of network resources decided at server end.** - *This relates to the path that will be used to move data from device to device or host to host. The client is usually not the best entity to decide which path is used for transmission.*

9. **Intelligent method of modifying files that exist only in non-random access storage systems.** - *Most systems copy the file to random access storage before the modification can begin. This is very inefficient if the file is to be replaced completely. If the client will be modifying the file in a sequential manner tape modification might be sufficient.*

10. **Clients can specify which networks and network transports they have available. This is reconfigurable on the fly.** - *In a distributed environment network components and hosts will, at some point, be unavailable due to scheduled and unscheduled outages. Network components should be able to dynamically update routing (network path) information without requiring a shutdown & reboot.*

11. **Can take storage systems out of service for PM or permanently without losing access to the other storage systems.** - *The distributed file system should not have a single point of failure. All components must support periodic maintenance.*

12. **Volume salvager, journals, and checkpoints.** - *The file system must support, sequential storage reclamation, system state restoration, and catastrophic failure recovery.*

13. **Backup and cloned volume support.** - *The file system should support the creation and management of backup and duplicate volumes for recovery and load balancing purposes. A volume is a large management unit consisting of a group of files or even an entire data-set.*

14. **Per-file access history.** - *A capability must exist to track file and volume access and utilization. This will, in part, be supported by the file system.*

15. **Should run on most, if not all, platforms on the market.** - *The best of all worlds is that the distributed file system selected by ECS runs on every platform in the world. In practice this will not be the case. The distributed file system should support the widest possible range of platforms.*

16. **Supports write-once, read-never storage systems (Salt-Mine).** - *The file system should allow for the possibility that directory and file entries will be made for files that will never be accessed. This is a file system scalability issue that relates to the number of files the system can effectively manage.*

17. **Support for parallel I/O.** - *If multiple paths are available from a host to a client, the file system should support data transmission over each path. This can be via a "packetizing" scheme or through some other method.*

18. **Support interface for heterogeneous network access.** - *If a heterogeneous network is required either from the standpoint of ingest or distribution, the file system must support either a translator, a client, or an API for on the fly translation.*

This page intentionally left blank.

Appendix B

Anticipated File Server Requirements

- *File Server should manage files. Unclear if HSM is better than other methods such as buffer caching.*
- *Unclear if Unix resident file manager would be more beneficial than separate file system managers.*
- *Inode structures are limiting but the alternative is to have no real insight into where data is being stored.*
- *An inode structure means that the file must be retrieved to the original location of its inode prior to movement.*

1. Operators should be able to manually alter the routing of File Server data sets to physical storage locations.
2. The File Server should be capable of generating and managing one or more backup copies of all stored data.
3. The File Server should be able to recover from database or file system corruptions and/or physical media failures without data loss.
4. The File Server should be capable of storing multiple versions of a single file.
5. The File Server should log all errors, and warnings in a central error log.
6. The File Server should be capable of performing normal operations without operator intervention.
7. The File Server should have multiple severity levels built in to error processing.
8. The File Server should alert the operator when specific errors or the number of errors reach a specific threshold.
9. The File Server should have some means of automatically testing the viability of data within the archive.
10. The File Server should be able to monitor the status, cost and performance of all storage systems in the archive.
11. The File Server should provide a mechanism for statistically monitoring the bit error rate (BER) of storage media in the archive.
12. The File Server should have a Master File Directory listing all files under its control.
13. Operations/systems personnel shall be able to access, list, or modify the contents of the MFD in a special privileged mode.
14. The File Server shall support industry standard communications and network protocols.

15. The File Server shall provide for continued performance, albeit in a degraded mode, when a device (disk or cartridge drive, operator’s console, etc.) fails.

16. Each File Server shall have tools that allow operations/systems/maintenance personnel to monitor performance, carry out maintenance, and alter operating parameters.

17. The File Server will provide some means of logically associating data.

18. The File Server shall support automated as well as manual migration of file between storage levels.

19. The File Server should provide some form of file traceability. - *(e.g. creation date, last modified date, UID/SID of the person accessing the data, etc.)*

20. The File Server should provide a manual means of moving files between levels in the storage hierarchy.

21. The File Server should support partial file retrievals.

22. The File Server should support multiple storage hierarchies. These should be configurable to the file level.

23. The File Server should support explicit, implicit and scheduled file retrieval and migration.

24. The File Server should retrieve files on demand.

25. The File Server should provide tunable migration strategies based on data type & class.

26. The File Server should provide a trashcan delete function.

27. The File Server should be able to export media for off-line storage or for import into other ATLs.

28. The File Server should provide self-describing media.

29. The File Server should support tape defragmentation and free space reclamation.

30. The File Server should provide a capability for periodic as well as threshold based viability testing for each medium under its control.

31. The File Server should support journaling and checkpointing of operations.

Appendix C

Anticipated Volume Serving Component Requirements

1. Does not sit on the data path between data and client. - Separate Control and Data Path exists. Data path does not include movement through the memory of the Volume Server.
2. Does not know anything about the data being managed. - A Volume Manager only needs to understand the locations of the media volumes, and the associated drives & ATLs. Specific information on volume contents is not needed.
3. The client provides the file serving mechanism that determines what data is recorded on each medium.
4. The client only needs to track the data location with a cross reference of the recorded files to a specific medium by volume name. - The client only needs the actual name of the volume to initiate a retrieval of a stored file.
5. The client also provides the drive interface to each drive. - The client is responsible for commanding the tape drive and positioning the tape to the appropriate file. The Volume Manager provides the client with the appropriate volume on an appropriate drive and data path.
6. The Volume Server provides the ability to accept predefined commands for media within its domain and executes those commands. - Individual commands can be strung together in script type files.
7. The Volume Server finds and moves media based only on the medium's logical name.
8. The Volume Server can manage media movement between archives.
9. The Volume Server can control multiple archive systems.
10. The Volume Server supports multiple form factors transparently.
11. The Volume Server manages both robotic and manual archives.
12. The Volume Server can run independently (without operator supervision).
13. The Volume Server will notify operator of serious problems.
14. All Volume Server commands and errors are logged in a central log.
15. The Volume Server must support media removal for remote processing (checkout).
16. The Volume Server must support media import of "foreign" media.
17. The Volume Server must be able to group media in logical categories, relations, classes, etc.
18. The Volume Server must insure volumes reside in only one logical grouping.
19. The Volume Server should accept logical grouping assignments from the client.

20. The Volume Server should support multi-tiered archive structures, and automatic and manual movement of media between these tiers as needed. Movement should be prioritized based on least recently mounted. Should be able to differentiate between media types.

21. The Volume Server should manage available media resources and inform operator when thresholds are reached. i.e. # of volumes in a class, total used, total remaining, add more media threshold.

22. The Volume Server allows specific pools of devices to be added and removed as needed by the system Administrator or the client. - Special purpose activities such as high interest processing can have a block of equipment dedicated to this task for a period of time.

23. The Volume Server should be scaleable. Either multiple copies working together or distributed control for a central storage pool.

24. The Volume Server software should be easily ported to various hosts.

25. The Volume Server control the availability of archive units for maintenance and diagnostics.

26. The Volume Server provides administration, database queries, manual load/unload and configuration capabilities.

27. The Volume Server should maintain statistics on soft and hard media errors.

28. The Volume Server should periodically or in some predetermined order, test the viability of data in the archive.

29. The Volume Server should recopy data that is reaching specific viability thresholds.

30. The Volume Server should support automated as well as manually mounted data volumes.

Abbreviations and Acronyms

ACL	Access Control List
AFS	Andrew File System
API	Applications Program Interface
ASCII	American Standard Code for Information Interchange
DCE	Distributed Computing Environment
DFS	Distributed File System
DME	Distributed Management Environment
ECS	EOSDIS Core System
EOS	Earth Observing System
FSMS	File Storage Management System
FTP	File Transfer Protocol
GUI	Graphical User Interface
NCSA	National Center for Supercomputing Applications
NFS	Network File System
NSL	National Storage Laboratory
OS	Operating System
OSF	Open Software Foundation
rcp	Remote Copy
RPC	Remote Procedure Call
UFS	Unix File System
uucp	Unix to Unix Copy
VFS	Virtual File System